

OfficeGRID - User's Manual

Last updated 19-11-2009 07:25

Contents

1	Introduction	2
2	Job handling	3
2.1	Graphical User Interface (GUI)	4
2.2	Command Line Interface (CLI)	5
3	Submit a job	5
3.1	GUI	5
3.2	CLI	7
3.2.1	og_submit	7
3.2.2	og_shcmd	9
3.2.3	og_job	11
4	Job status	13
4.1	GUI	13
4.2	CLI	15
4.2.1	List Job Status	15
4.2.2	List Job Results	16
4.2.3	Retrieve a Job Result	17
4.2.4	Remove Submitted Jobs	18
5	Resource Reservation	19
5.1	Job Reservations	19
6	Python API	20
7	Configuration	21
7.1	Configuration Wizard	21
7.2	Manual Configuration Changes	27

1 Introduction

OfficeGRID from MESH Technologies[1] is a high performance Grid middle-ware which brings Grid technologies to everyone. With OfficeGRID is simple and easy to harness the excess power of the computers available at locations ranging from small offices to entire organisations and companies.

OfficeGRID runs on all major computing platforms:

- Microsoft Windows[2]
- Linux[3]
- Mac OS X[4]

OfficeGRID consists of the following components:

Server Each OfficeGRID includes a server, which should be available at all times. Users contact the server whenever they want to submit new jobs, query Grid or job status, retrieve completed job output, remove old jobs, and so on. A server handles job file storage and job queues and generally takes care of the job flow in the Grid.

Executor A server controls a number of executors, that take care of the actual job execution. When an executor is idle it requests a suitable job from the server, and if such a job is waiting in the job queue, it is dispatched to the executor. It is then executed and results are handed back to the server. If no suitable jobs are available, the executor will wait a while before requesting a job again. Executors can be bound to serve only a subset of the job queues for more fine grained control. Additionally an executor can be identified by platform and tags. Platform can be used to make sure that Windows specific jobs end up on Windows based executors, Linux jobs on Linux based executors and so on, if the Grid contains multiple executor platforms. Tags are normally used for identifying executors with specific run-time environments, which may be a particular application like Java, etc.

User Interfaces There are two user interfaces for OfficeGRID interaction - a graphical interface for user friendly job handling and a set of command line applications for efficient/scripted job handling.

Web interface OfficeGRID installations may include a web server that allows administrators and possibly users to monitor and interact with OfficeGRID through a web interface.

2 Job handling

OfficeGRID jobs can be handled in two ways: using a GUI interface or with a collection of command line tools. Choose the method you prefer or a combination of both. First we will go through some of the job properties that both methods share.

When a client wants to retrieve information about a job, the client must somehow identify the job of interest. A job can be identified by the following properties:

id A unique identification number.

queue A job is always submitted to a particular queue which may have any number of executors associated with it.

tag A job can require one or more executor tags as a way of requesting a specific feature or environment.

group A job can be marked as belonging to one or more groups of jobs in order to make later identification easier.

flag A job in a group can be assigned a flag to distinguish it from other jobs in the same group.

owner The username that the job was submitted with.

password The password for the account to run jobs under, i.e. the password for the job owner mentioned above.

Please note that **owner** refers to the user account to run jobs under, i.e. the user login on the **executor** host running the job. Thus if you have an account with, say, username *john* and password *johndoe* on the executor, you should similarly set job owner to *john* and password to *johndoe*, even if your login on the client host is different.

It should also be noted that on some platforms the local security policy prevents signing in with an empty password.

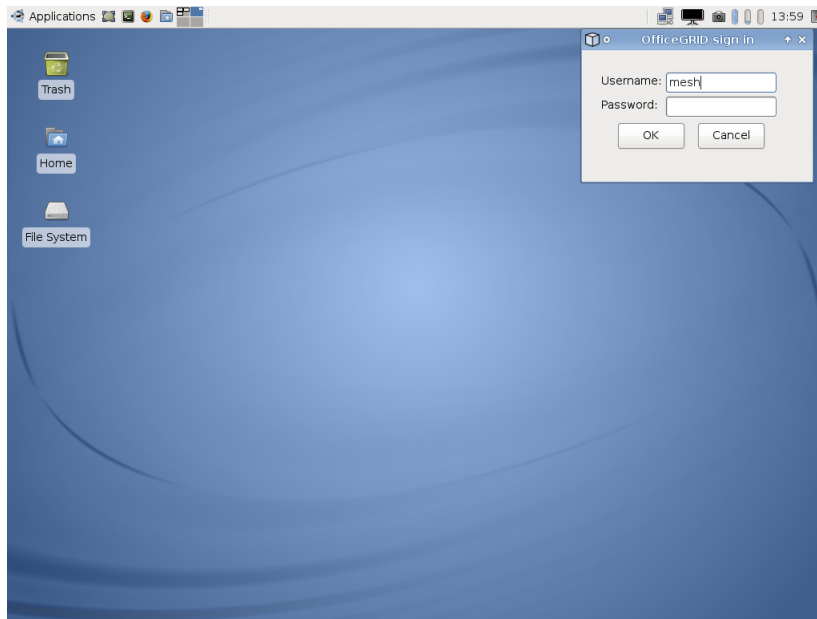
When jobs are submitted to OfficeGRID they can be guided to be run on a particular operating system.

platform The name of the operating systems that can run the job.

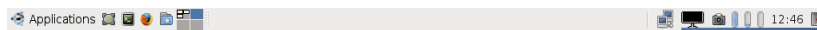
- windows
- linux
- mac
- all

2.1 Graphical User Interface (GUI)

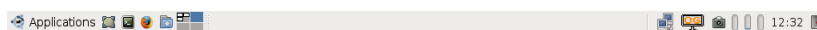
OfficeGRID includes a graphical user interface in the form of a system tray icon with associated windows. Start up the GUI from the application menu or by running the ogtaskbar command from a command line. Now the OfficeGRID tray icon will appear (in the system tray adjacent to the clock) along with a sign in dialog. The graphical client provides single sign on through the sign in dialog, so after signing in all further operations will use the signed in user in all job owner fields.



Until a user is signed in, the tray icon will display a blank screen icon.



When a user signs in the tray icon will change to display a screen with a small OG logo.



At all times hovering the cursor over the icon will display a short sign in status message.



After signing in with the username and password, right click on the taskbar icon to access the window menu.



Now choose **Submit job** or **Job status** to open the corresponding window. The associated windows will be further described below. To sign out or sign in as another user simply select Sign On/Off. Choosing **Cancel** in the sign in dialog effectively signs off the current user.

2.2 Command Line Interface (CLI)

OfficeGRID additionally supplies a set of command line applications which may prove more handy and efficient than the GUI client when handling a large number of jobs. Each of the command line applications can be run from a shell/prompt.

3 Submit a job

Submitting a job will be described below for the GUI and CLI clients.

3.1 GUI

Open the submit job window from the tray icon, and fill in the details for the job. The submit job dialog will look something like this:

OfficeGRID - Submit job

Submit one or more command lines to OfficeGRID

echo 'hello grid'

Owner: mesh Shell: all No. of copies: 1

CPU cores: 1 Disk (MB): 0 Memory (MB): 1

Network (Mb): 0 Walltime (s): 3600 Sched. priority: 0

Name of the queue to submit to: default

Hostname of job executor (empty for any):

Send file along with the job when submitting. Checked files are cached for future use.

Send: []

Retrieve these file(s) as part of the job result.

Retrieve: []

Run job on a machine with the given tag(s). E.g. only run job in a specific environment.

Tag: []

Specify one or more executor flags to match.

Flag: []

Group name to match.

Group: []

☐ Show job ID's upon submit

Exit Commit

15:45:24 Ready

Please note that the Owner button will display the name of the current system user if not signed in and the name of the signed in user otherwise. In the above example the name of the signed in user is **mesh**.

The simplest of valid jobs is the 'hello grid' job, which is submitted by writing/leaving:

```
echo hello grid
```

in the command field in the top left corner of the window and clicking the **Commit** button in the bottom right corner. Upon submitting the job the status bar in the bottom of the window will be updated with the submit status as emphasised with the red square below:

OfficeGRID - Submit job

Submit one or more command lines to OfficeGRID

echo 'hello grid'

CPU cores: 1, Disk (MB): 0, Memory (MB): 1

Name of the queue to submit to: default

Owner: mesh, Shell: all, No. of copies: 1

Network (Mb): 0, Walltime (s): 3600, Sched. priority: 0

Hostname of job executor (empty for any):

Send file along with the job when submitting. Checked files are cached for future use.

Retrieve these file(s) as part of the job result.

Run job on a machine with the given tag(s). E.g. only run job in a specific environment.

Specify one or more executor flags to match.

Group name to match.

☐ Show job ID's upon submit

Exit Commit

15:45:47 Submitted 1 job(s)

Further details about the job execution progress is available in the Job status window as described below.

For more advanced jobs the other fields in the submit window will be needed, but most simple jobs only require changes in the topmost row of fields.

3.2 CLI

From the command line run one of the three submit commands. The submit command of choice depends on the format of the job, e.g. if it is a single command, a script or a more complex job description. Inexperienced users of OfficeGRID will probably prefer to start out with the GUI client or the `og_shcmd` CLI command.

3.2.1 `og_submit`

The `og_submit` command is used to submit batch jobs in the form of a job description to OfficeGRID:

```
Usage: og_submit [options] <job description>
Submit job specified in job description file to OfficeGRID.
The job description is a file containing e.g.:
```

```
main: echo
platform: all
copies: 2
```

args: first command ;; second command

The above example will create two jobs running the "echo" command with arguments "first command" and "second command" respectively and the jobs can be executed on all platforms. Thus the two jobs will produce an output of "first command" and "second command".

The following options can be used:

-a, --askpw	Prompt for a password to use in the job or/request, i.e. password for the owner.
-C, --cores <CPU cores>	Number of CPU cores required by the job (integer or keyword).
-c, --copies <number>	The number of copies of the job.
-D, --disk <megabytes>	Megabytes of disk required by the job (integer or keyword).
-d, --deposit <file>	Send file as part of the job when the job is submitted. The file is cached for future references.
-f, --flag <job flag>	Specify a flag to match.
-g, --group <group label>	A group label that can be used to mark and identify related jobs.
-h, --help	Displays this help message.
-I, --inform <notify data>	Inform/notify about job state changes. The notify data is a colon separated list of protocol, state and recipient addresses. The data "email:done:you@company.com" will result in an email to you@company.com when the job is done.
-i, --id	Current supported protocols: ['email', 'jabber'] Return job id for submissions and match id for inquiries.
-m, --memory <megabytes>	Megabytes of memory required by the job (integer or keyword).
-N, --network <megabits>	Megabits of network bandwidth required by the job (integer or keyword).
-n, --host <hostname>	Is used to target the a specific machine, running an executor. This may be a hostname or an IP address.
-P, --priority <number>	The scheduling priority of the job. A higher value means greater priority and thus increased scheduling chance.
-p, --platform <platform>	The platform(s) that the job is allowed to be executed on.

<code>-q, --queue <queuename></code>	The queue <code><queuename></code> that is matched to. If this option is not used the default queue will be used for submission and all queues for inquiries.
<code>-r, --retrieve <path></code>	Marks file with given path as a result that should be saved as part of the job output when used in job submit commands. Specifies the directory path where results are to be saved when used in result retrieval (<code>og_res</code>).
<code>-s, --send <file></code>	Send file as part of the job when the job is submitted.
<code>-t, --tag <tag string></code>	Is used to find machine that matches the given tag. The machine tag is set as an option to the job wrapper.
<code>-u, --user <username></code>	The job owner <code><username></code> that is either matched to or used.
<code>-w, --walltime <time></code>	Wall clock time required by the job. Time is in seconds or "w:d:h:m:s"-format where w is weeks, d is days, h is hours, m is minutes and s is seconds. Also accepts suffix format strings like "m:s", "h:m:s" or "d:h:m:s". Thus "3:20" means 3 minutes and 20 seconds. Also allows MAX or UNLIMITED keyword.

Copyright 2003 - 2008 (c) MESH-Technologies A/S.

3.2.2 `og_shcmd`

The command `og_shcmd` is used to submit one-line batch jobs to OfficeGRID. `og_shcmd` is used in the following way:

Usage: `og_shcmd [options] <command line>`

Submit command line job to OfficeGRID and restrict the job to be run on executors of the specified platform.

The following options can be used:

<code>-a, --askpw</code>	Prompt for a password to use in the job or/request, i.e. password for the owner.
<code>-C, --cores <CPU cores></code>	Number of CPU cores required by the job (integer or keyword).
<code>-c, --copies <number></code>	The number of copies of the job.
<code>-D, --disk <megabytes></code>	Megabytes of disk required

-d, --deposit <file>	by the job (integer or keyword). Send file as part of the job when the job is submitted. The file is cached for future references.
-f, --flag <job flag>	Specify a flag to match.
-g, --group <group label>	A group label that can be used to mark and identify related jobs.
-h, --help	Displays this help message.
-I, --inform <notify data>	Inform/notify about job state changes. The notify data is a colon separated list of protocol, state and recipient addresses. The data "email:done:you@company.com" will result in an email to you@company.com when the job is done. Current supported protocols: ['email', 'jabber']
-i, --id	Return job id for submissions and match id for inquiries.
-m, --memory <megabytes>	Megabytes of memory required by the job (integer or keyword).
-N, --network <megabits>	Megabits of network bandwidth required by the job (integer or keyword).
-n, --host <hostname>	Is used to target the a specific machine, running an executor. This may be a hostname or an IP address.
-P, --priority <number>	The scheduling priority of the job. A higher value means greater priority and thus increased scheduling chance.
-p, --platform <platform>	The platform(s) that the job is allowed to be executed on.
-q, --queue <queuenam>	The queue <queuenam> that is matched to. If this option is not used the default queue will be used for submission and all queues for inquiries.
-r, --retrieve <path>	Marks file with given path as a result that should be saved as part of the job output when used in job submit commands. Specifies the directory path where results are to be saved when used in result retrieval (og_res).
-s, --send <file>	Send file as part of the job when the job is submitted.
-t, --tag <tag string>	Is used to find machine that matches the given tag. The machine tag is set as an

	option to the job wrapper.
-u, --user <username>	The job owner <username> that is either matched to or used.
-w, --walltime <time>	Wall clock time required by the job. Time is in seconds or "w:d:h:m:s"-format where w is weeks, d is days, h is hours, m is minutes and s is seconds. Also accepts suffix format strings like "m:s", "h:m:s" or "d:h:m:s". Thus "3:20" means 3 minutes and 20 seconds. Also allows MAX or UNLIMITED keyword.

Copyright 2003 - 2008 (c) MESH-Technologies A/S.

As an example the 'hello grid' job mentioned in the GUI client section could be submitted from the command line with the command:

```
og_shcmd echo hello grid
```

3.2.3 og_job

The command `og_job` is a variant of `og_shcmd` which may be simpler to use when the job should just run a script with some input. `og_job` is used in the following way:

```
usage: og_job [options] <job script> [job script arguments]
Submit a script to the OfficeGRID system.
og_job is a convenience tool used for submitting just a script,
for all other uses og_shcmd or og_submit may be more convenient.
The following options can be used:
```

-a, --askpw	Prompt for a password to use in the job or/request, i.e. password for the owner.
-C, --cores <CPU cores>	Number of CPU cores required by the job (integer or keyword).
-c, --copies <number>	The number of copies of the job.
-D, --disk <megabytes>	Megabytes of disk required by the job (integer or keyword).
-d, --deposit <file>	Send file as part of the job when the job is submitted. The file is cached for future references.
-f, --flag <job flag>	Specify a flag to match.

-g, --group <group label>	A group label that can be used to mark and identify related jobs.
-h, --help	Displays this help message.
-I, --inform <notify data>	Inform/notify about job state changes. The notify data is a colon separated list of protocol, state and recipient addresses. The data "email:done:you@company.com" will result in an email to you@company.com when the job is done. Current supported protocols: ['email', 'jabber']
-i, --id	Return job id for submissions and match id for inquiries.
-m, --memory <megabytes>	Megabytes of memory required by the job (integer or keyword).
-N, --network <megabits>	Megabits of network bandwidth required by the job (integer or keyword).
-n, --host <hostname>	Is used to target the a specific machine, running an executor. This may be a hostname or an IP address.
-P, --priority <number>	The scheduling priority of the job. A higher value means greater priority and thus increased scheduling chance.
-p, --platform <platform>	The platform(s) that the job is allowed to be executed on.
-q, --queue <queuenam>	The queue <queuenam> that is matched to. If this option is not used the default queue will be used for submission and all queues for inquiries.
-r, --retrieve <path>	Marks file with given path as a result that should be saved as part of the job output when used in job submit commands. Specifies the directory path where results are to be saved when used in result retrieval (og_res).
-s, --send <file>	Send file as part of the job when the job is submitted.
-t, --tag <tag string>	Is used to find machine that matches the given tag. The machine tag is set as an option to the job wrapper.
-u, --user <username>	The job owner <username> that is either matched to or used.
-w, --walltime <time>	Wall clock time required by the job. Time is in seconds or "w:d:h:m:s"-format where w is weeks, d is days, h is hours,

m is minutes and s is seconds.
Also accepts suffix format strings like
"m:s", "h:m:s" or "d:h:m:s". Thus "3:20"
means 3 minutes and 20 seconds.
Also allows MAX or UNLIMITED keyword.

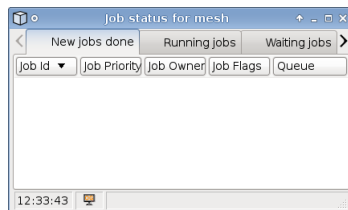
Copyright 2003 - 2008 (c) MESH-Technologies A/S.

4 Job status

If you followed the steps in above, the jobs should now be submitted to your Grid. So, the next natural step would be to check the status of the jobs.

4.1 GUI

Right click on the taskbar icon and choose **Job status** to get the following dialog:



It consist of four tabs:

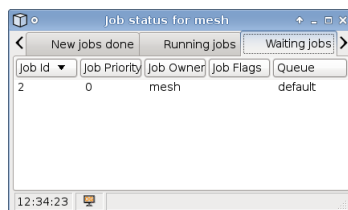
New jobs done List of finished jobs not yet retrieved.

Running jobs List of currently running jobs.

Waiting jobs List of jobs not yet started.

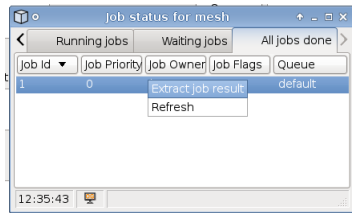
All jobs done List of all finished jobs.

When jobs are submitted they will show up in the corresponding status tabs.
E.g. new jobs will be visible in the Waiting tab until they start executing:

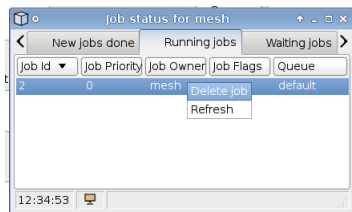


A right click menu is associated with every item on the lists, the menu has some of the following actions:

Extract job result Results in a job output dialog where it is possible to select the location where the files associated with the job should be written. Any stdout and stderr output of the job is printed in a new dialog. This option is available for finished jobs.



Delete job Deletes the selected job - this option only applies to unfinished jobs.



Refresh Refresh the current list.

Selecting Extract job result on a finished job will open a directory selection dialog. Any job output files will be retrieved from the Grid and stored in the chosen directory. Additionally a new window containing the job command line output will be opened. In the case of the hello grid command line job this will be:



The number of visible tabs in the Job status window depends on the window size. If the window is too small to display all four tabs, the window will instead include arrow buttons in each side at the top of the window. These arrow buttons can be used to navigate to the remaining tabs without resizing the window.

4.2 CLI

The same features are available from the command line through the use of four commands.

4.2.1 List Job Status

Use the `og_stat` command to get status of one or more submitted jobs:

usage: `og_stat` [options]

Show status of job(s) in the OfficeGRID system.

The following options can be used:

<code>-a, --askpw</code>	Prompt for a password to use in the job or/request, i.e. password for the owner.
<code>-C, --cores <CPU cores></code>	Number of CPU cores required by the job (integer or keyword).
<code>-D, --disk <megabytes></code>	Megabytes of disk required by the job (integer or keyword).
<code>-f, --flag <job flag></code>	Specify a flag to match.
<code>-g, --group <group label></code>	A group label that can be used to mark and identify related jobs.
<code>-h, --help</code>	Displays this help message.
<code>-i, --id</code>	Return job id for submissions and match id for inquiries.
<code>-N, --network <megabits></code>	Megabits of network bandwidth required by the job (integer or keyword).
<code>-n, --host <hostname></code>	Is used to target the a specific machine, running an executor. This may be a hostname or an IP address.
<code>-m, --memory <megabytes></code>	Megabytes of memory required by the job (integer or keyword).
<code>-P, --priority <number></code>	The scheduling priority of the job. A higher value means greater priority and thus increased scheduling chance.
<code>-p, --platform <platform></code>	The platform(s) that the job is allowed to be executed on.
<code>-q, --queue <queue name></code>	The queue <queue name> that is matched to. If this option is not used the default queue will be used for submission and all queues for inquiries.
<code>-t, --tag <tag string></code>	Is used to find machine that matches the given tag. The machine tag is set as an

-u, --user <username> -w, --walltime <time>	option to the job wrapper. The job owner <username> that is either matched to or used. Wall clock time required by the job. Time is in seconds or "w:d:h:m:s"-format where w is weeks, d is days, h is hours, m is minutes and s is seconds. Also accepts suffix format strings like "m:s", "h:m:s" or "d:h:m:s". Thus "3:20" means 3 minutes and 20 seconds. Also allows MAX or UNLIMITED keyword.
---	--

Copyright 2003 - 2008 (c) MESH-Technologies A/S.

If you already submitted a job it will show up on the list printed by the command:

og_stat

If the status of a particular job is wanted it can be exclusively selected by supplying the -i JOBID option, where JOBID is the ID given to the job during job submission.

4.2.2 List Job Results

After jobs have been executed on machines running the executor, the results from the executions can be listed by using the `og_reslist` command:

usage: `og_reslist` [options]

Returns a list of finished jobs from the OfficeGRID system.

The following options can be used:

-a, --askpw -f, --flag <job flag> -g, --group <group label> -h, --help -i, --id -l, --limit <status>	Prompt for a password to use in the job or/request, i.e. password for the owner. Specify a flag to match. A group label that can be used to mark and identify related jobs. Displays this help message. Return job id for submissions and match id for inquiries. Limit list to jobs with specified status in the queue. Supported values
--	---

	are new died terminating running done Done+Fetch
-n, --host <hostname>	Is used to target the a specific machine, running an executor. This may be a hostname or an IP address.
-P, --priority <number>	The scheduling priority of the job. A higher value means greater priority and thus increased scheduling chance.
-q, --queue <queue name>	The queue <queue name> that is matched to. If this option is not used the default queue will be used for submission and all queues for inquiries.
-t, --tag <tag string>	Is used to find machine that matches the given tag. The machine tag is set as an option to the job wrapper.
-u, --user <username>	The job owner <username> that is either matched to or used.

Copyright 2003 - 2008 (c) MESH-Technologies A/S.

4.2.3 Retrieve a Job Result

After a job has been executed the result can be found by running the og_res command:

usage: og_res [options]

Retrieves a result from the OfficeGRID system.

The following options can be used:

-a, --askpw	Prompt for a password to use in the job or/request, i.e. password for the owner.
-b, --block	Block until the result is ready.
-e, --err <file>	Write stderr to <file>.
-f, --flag <job flag>	Specify a flag to match.
-g, --group <group label>	A group label that can be used to mark and identify related jobs.
-h, --help	Displays this help message.
-i, --id	Return job id for submissions and match id for inquiries.
-n, --host <hostname>	Is used to target the a specific machine, running an executor. This may be a hostname or an IP address.
-o, --out <file>	Write stdout to <file>.
-q, --queue <queue name>	The queue <queue name> that is matched to. If this option is not used

<code>-r, --retrieve <path></code> <code>-t, --tag <tag string></code> <code>-u, --user <username></code>	the default queue will be used for submission and all queues for inquiries. Marks file with given path as a result that should be saved as part of the job output when used in job submit commands. Specifies the directory path where results are to be saved when used in result retrieval (og_res). Is used to find machine that matches the given tag. The machine tag is set as an option to the job wrapper. The job owner <username> that is either matched to or used.
--	---

Copyright 2003 - 2008 (c) MESH-Technologies A/S.

4.2.4 Remove Submitted Jobs

The og_jobrm command is executed in the following way:

usage: og_jobrm [options]

Removes a job from the OfficeGRID system.

The following options can be used:

<code>-a, --askpw</code> <code>-f, --flag <job flag></code> <code>-g, --group <group label></code> <code>-h, --help</code> <code>-i, --id</code> <code>-n, --host <hostname></code> <code>-q, --queue <queue name></code> <code>-t, --tag <tag string></code> <code>-u, --user <username></code>	Prompt for a password to use in the job or/request, i.e. password for the owner. Specify a flag to match. A group label that can be used to mark and identify related jobs. Displays this help message. Return job id for submissions and match id for inquiries. Is used to target the a specific machine, running an executor. This may be a hostname or an IP address. The queue <queue name> that is matched to. If this option is not used the default queue will be used for submission and all queues for inquiries. Is used to find machine that matches the given tag. The machine tag is set as an option to the job wrapper. The job owner <username> that is either matched to or used.
---	--

For example all the jobs in the job group `my_job_group`, from the default queue, can be removed by issuing the command:

```
og_jobrm -g my_job_group
```

5 Resource Reservation

In recent versions of OfficeGRID, jobs and executors support resource reservations and limits. Executors can specify the available amount of system resources like CPU cores, memory, disk, network and wall time. Likewise jobs can request a specific amount of each of the resources as a way of booking those resources on the executor where they run. The OfficeGRID server takes care of only handing out jobs to executors that provide the requested resources. Thus if an executor is configured to use four CPU cores, the server will only hand out jobs to it until all four cores are reserved or until another resource is depleted. As an example the executor may be assigned one dual core job and two single core jobs to run concurrently. When one of the jobs finishes, the executor may be able to take a new job if the remaining free resources allow it.

5.1 Job Reservations

Executors may be configured with job limits, so in order to get your job executed at all it may be necessary to specify reasonable job reservation values (cores, disk, memory, network and walltime). The first four are voluntary ones that are used to prevent overbooking the executors. All reservations can take positive integer values as well as the keywords `MAX` and `UNLIMITED`. If a reservation is set to `MAX`, that particular reservation will always be translated on the fly to the maximum available value for the first available and matching execution request. Thus if `cores` is set to `MAX` in the job and the job is about to be scheduled for execution on an executor with 4 cores, the job reservation will be dynamically changed to 4 at that point in time. If a reservation field is set to `UNLIMITED`, the job will only be executed on executors that explicitly disabled limits on that field. Resource utilisation and job scheduling both improve if job reservations are reasonably accurate, so it is highly recommended to provide reasonable values. Walltime is a strict limit in the sense that jobs will be killed if they exceed

the reserved amount of time. This may lead to the tempting choice of specifying UNLIMITED walltime, but this is not generally a good idea, because it will most likely limit the number of suitable executors and because it significantly reduces scheduling efficiency. It is highly recommended to always set a walltime reservation, but possibly a high one if the actual time needed is unknown. Setting walltime to MAX may be a feasible alternative to UNLIMITED unless the job requires more time than the lowest allowed wall time for any suitable executor.

6 Python API

OfficeGRID includes a Python[7] module with bindings to all the client operations. This makes programming with OfficeGRID extremely flexible without adding unnecessary complexity. The Python programming language is easy to learn, yet very powerful, so with this module there's no limit to the possibilities of advanced job and Grid interaction.

When the module is installed it is just a matter of importing it in your python application with the line:

```
import officegrid.client
```

Single sign on is possible with the `sign_in` function:

```
officegrid.client.sign_in('myuser', 'mypassword')
```

Now the included Job and Grid objects can be instantiated with the lines:

```
job = officegrid.client.Job()
grid = officegrid.client.Grid()
```

A quick job execution example could be:

```
job.set('main', 'echo %s' % "hello grid")
job.submit()
print job.status()
job.result('.')
```

Similarly all kinds of Grid information may be queried like this:

```
print grid.status()
print grid.queues()
print grid.executing()
print grid.results()
print grid.statistics()
```

Finally sign off is possible with the `sign_off` function:

```
officegrid.client.sign_off()
```

Please refer to the sample `ogtest.py` program included in the installation for more detailed examples of using the Job and Grid objects.

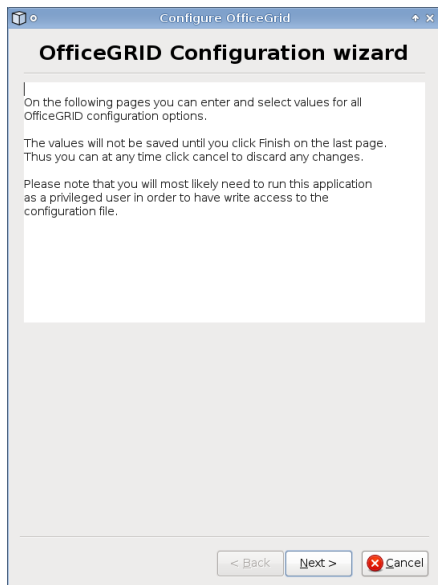
7 Configuration

The base installation includes a simple local server, executor and client configuration. In case a more advanced setup is desired, it is necessary to change the OfficeGRID configuration. This can be done by using the configuration wizard or by manually editing the configuration file, `officegrid.cfg`.

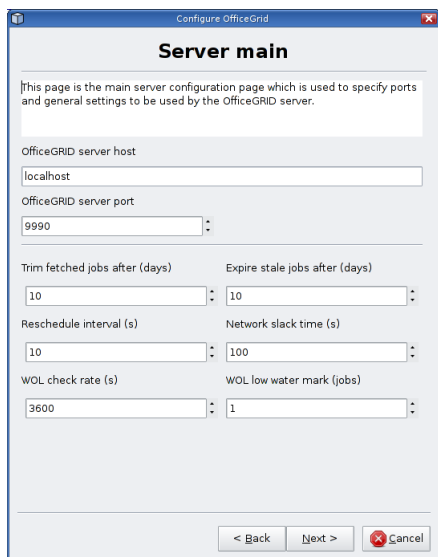
7.1 Configuration Wizard

Open the configuration wizard, which takes you through a number of configuration questions and finally updates the OfficeGRID configuration according to your replies. The wizard can be started from the application menu or by running the `ogconfwiz` command from the command line. Depending on the installation and platform some or all of the below pages may be accessible.

The intro page gives a few instructions for newcomers. Simply click **next** to proceed:



The server page includes general server options:



The server notification page contains server options related to job notification:

Server Notify

This page is the server notification configuration page which is used to specify addresses and other notification settings to be used by the OfficeGRID server. Notification settings are used by the message transports when informing users about status changes for their jobs.

E-mail notification

SMTP server: localhost SMTP port: 25

SMTP username: officegrid SMTP password:

☐ Secure SMTP session with TLS

Jabber notification

Jabber server: jabber.org Jabber port: 5222

Jabber username: officegrid Jabber password:

☐ Secure Jabber session over SSL

< Back Next > Cancel

The server paths page contains file specific server options:

Server paths

Files, paths and logging settings for the OfficeGRID server.

The location of the runtime files (install dir): /usr/local/og Browse

The directory with the license file (og.lic): /usr/local/og Browse

Temporary dir: /tmp Browse

Where to write the log. Leave empty to disable log: /tmp/server.log Browse

How detailed the log should be: info

Automatically rotate log when it reaches a certain size in bytes: 1048576

Keep this many old log files in the rotation scheme: 30

< Back Next > Cancel

The executor page includes general executor options:

Configure OfficeGrid

Executor main

Executor specific configuration parameters.
The WOL address may be an explicit MAC address or use the AUTO keyword for auto detection. Leave it empty to disable WOL for this host.

Job poll rate (s): 20 | License renew rate (s): 3600 | Kill monitor rate (s): 15

Queues to accept job from: default | Tags to identify the executor:

Flags to mark the executor: | Group of jobs to accept:

Total resources available on this executor:

CPU cores: 1	Memory (MB): 1024	Platform: linux
Disk (MB): 8192	Network (Mb): 1000	Walltime: 2592000

WOL MAC address to send wake up packets to if executor is offline but needed:

< Back | Next > | Cancel

The executor paths page contains file specific executor options:

Configure OfficeGrid

Executor paths

Files, paths and logging settings for the OfficeGRID executor

Tmp dir: Where to store temporary files
/tmp | Browse

Log file: Where to write the log. Leave empty to disable log
/tmp/executor.log | Browse

How detailed the log should be
info

Automatically rotate log when it reaches a certain size in bytes
1048576

Keep this many old log files in the rotation scheme
30

Run dir: The location of the runtime files (install dir)
/usr/local/log | Browse

< Back | Next > | Cancel

The web page includes general web interface options:

Web

Settings for the OfficeGRID web server

OfficeGRID web server host
localhost

OfficeGRID web server port
9991

OfficeGRID web server user/group (UNIX only feature)
nobody nobody

Number of days before cleaning temporary download files
14

< Back Next > Cancel

The web paths page contains file specific web interface options:

Web paths

Files, paths and logging settings for the OfficeGRID web server

Run dir: The location of the runtime files (install dir)
/usr/local/log Browse

Tmp dir: Where to store temporary files
/tmp Browse

Log file: Where to write the log. Leave empty to disable log
/tmp/webadmin.log Browse

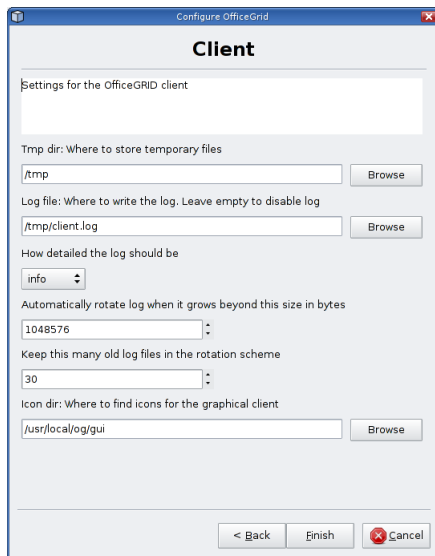
How detailed the log should be
info

Automatically rotate log when it reaches a certain size in bytes
1048576

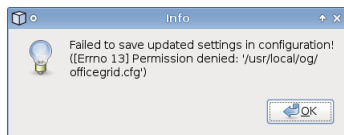
Keep this many old log files in the rotation scheme
30

< Back Next > Cancel

The client page contains all client options and a **finish** button instead of a **next** button. No changes are saved before you click **finish**, so you may use the **back** and **next** buttons to navigate back and forth through the pages as much as you want before finally saving the changes or discarding them by clicking **cancel**.



In some cases saving the configuration changes may result in an error message. This is most often a matter of write permissions to the configuration file.



So if you get an error like the one above, please run the wizard again logged in as a user with write permissions on the file (Hint: in most cases you need administrative rights).

”Important:” On some platforms including Windows Vista it may not even yield an error if the configuration wizard is running as a user without write permissions on the configuration file, but the correct file simply won’t be updated! If this is the case, please explicitly run the configuration wizard with administrator privileges in order to actually modify the real configuration. Right clicking the configuration wizard icon ”may” provide such an option, but otherwise please refer to the documentation for your particular operating system for further details about running commands with administrator privileges.

As seen above each field in the wizard has a short description, but please feel free to contact MESH in case it isn’t clear what they mean.

7.2 Manual Configuration Changes

Open up the OfficeGRID configuration file, `officegrid.cfg`, in your favourite editor and change the settings. It uses a 'ini' file like structure where `[NAME]` represents a section and `'#'` marks the rest of the line as a comment. Each of the options in the configuration is accompanied by a short description, so hopefully most options are self-explanatory.

The configuration file is separated in to the following sections:

- Default
- Server
- Executor
- Client

7.3 Client Configuration

As a user you should only worry about the Client section, which includes three variables:

TMPDIR

Path pointing to a directory where the client should write temporary files.

WRITE_LOG

Path pointing to the client log file.

ICON_DIR

Path to the directory holding the GUI client icons. This is normally the "gui" directory located in the OfficeGRID installation directory.

On Linux hosts this section could look like this:

```
[CLIENT]
TMPDIR = /tmp/OGclient
WRITE_LOG = /tmp/OGclient/client.log
ICON_DIR = /usr/local/og/gui
```